**OPERATING SYSTEM Project**

***Airline Reservation System***

**Group Members: SE**

**Div: B**

**Batch: B1**

|  |  |
| --- | --- |
| **NAME** | **ROLL NO.** |
| **Anagha Patil** | **408** |
| **Chinmay Patil** | **409** |
| **Aakansha Patil** | **407** |
| **Simran Pawar** | **415** |
| **Raj Pawar** | **414** |

AIM :

TO DEVELOP AIRLINES RESERVATION SYSTEM USING FIRST COME FIRST SERVE ALGORITHM (FCFS)

THEORY:

## **First Come First Serve (FCFS)**

            In the "First come first serve" scheduling algorithm, as the name suggests, the process which arrives first, gets executed first, or we can say that the process which requests the CPU first, gets the CPU allocated first.

A perfect real life example of FCFS scheduling is buying tickets,

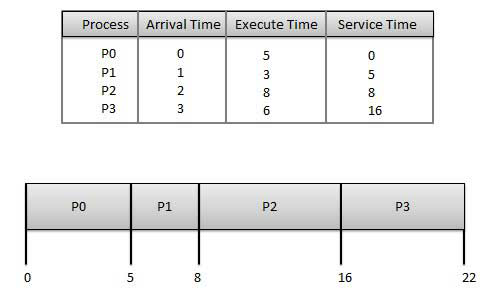
Talking about our project reservation system is based on concept of reserving airline tickets by providing with your names and phone numbers and the preferred seat number so the system reserves the seat for you and later that particular seat is not shown available to other user.

  First Come First Serve, is just like FIFO(First in First out) Queue data structure, where the data element which is added to the queue first, is the one who leaves the queue first.

This is used in Batch Systems.

        It's easy to understand and implement programmatically, using a Queue data structure, where a new process enters through the tail of the queue, and the scheduler selects process from the head of the queue.

* Jobs are executed on first come, first serve basis.
* It is a non-preemptive, pre-emptive scheduling algorithm.
* Easy to understand and implement.
* Its implementation is based on FIFO queue.
* Poor in performance as average wait time is high.



Wait time of each process is as follows −

|  |  |
| --- | --- |
| **Process** | **Wait Time : Service Time - Arrival Time** |
| P0 | 0 - 0 = 0 |
| P1 | 5 - 1 = 4 |
| P2 | 8 - 2 = 6 |
| P3 | 16 - 3 = 13 |

Average Wait Time: (0+4+6+13) / 4 = 5.75

ALGORITHM :

1-  Input the processes along with their burst time     (bt).

2-  Find waiting time (wt) for all processes.

3-  As first process that comes need not to wait so

    waiting time for process 1 will be 0 i.e. wt[0] = 0.

4-  Find waiting time for all other processes i.e. for

     process i ->

       wt[i] = bt[i-1] + wt[i-1] .

5-  Find turnaround time = waiting\_time + burst\_time

    for all processes.

6-  Find average waiting time =

                 total\_waiting\_time / no\_of\_processes.

7-  Similarly, find average turnaround time =

                 total\_turn\_around\_time / no\_of\_processes.

SOURCE CODE :

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

typedef struct CustomerDetails //STRUCTURE DECLARATION

{

char name[20];

char phonenumber[15];

char seats[20];

}Customer;

Customer c[100];

char arr[10],dep[10];

char seats[101][10]={"E","E","E","E","E","E","E","E","E","E",

"E","E","E","E","E","E","E","E","E","E",

"E","E","E","E","E","E","E","E","E","E",

"E","E","E","E","E","E","E","E","E","E",

"E","E","E","E","E","E","E","E","E","E",

"E","E","E","E","E","E","E","E","E","E",

"E","E","E","E","E","E","E","E","E","E",

"E","E","E","E","E","E","E","E","E","E",

"E","E","E","E","E","E","E","E","E","E",

"E","E","E","E","E","E","E","E","E","E","E"};

void welcome()

{

printf(" -------------------------------------------------------------------------\n");

printf("| |\n");

printf("| |\n");

printf("| OOOOOO OOOOOO OOOOOO OOOOOO OOOOOO OOOOOO O O OOOOOOO OOOOOO |\n");

printf("| O O O O O O O O O O O O |\n");

printf("| O OOOOO OOOOOO OOOOO OOOOO O O O O O O OOOOO OOOOOO |\n");

printf("| O O O O O O O O O O O O O O |\n");

printf("| OOOOOO O O OOOOOO OOOOOO O OOOOOO O O O OOOOOO OOOOOO |\n");

printf("| |\n");

printf(" -------------------------------------------------------------------------\n");

printf("\t\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\t\t\* \*\n");

printf("\t\t\* ----------------------------- \*\n");

printf("\t\t\* WELCOME TO BINGO AIRLINES \*\n");

printf("\t\t\* ----------------------------- \*\n");

printf("\t\t\* \*\n");

printf("\t\t\* \*\n");

printf("\t\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n\n");

}

void seat\_arrangement()

{

int i,j,index=0;

printf("\n");

for(i=0; i<10; i++)

{

printf("\t\t");

for(j=0; j<10; j++)

{

printf("%d.%s\t",index+1,seats[index+1]);

index++;

}

printf("\n");

}

}

void display(int n,int ch)

{

int i;

printf("\n\t\tE: SEAT EMPTY\t\t\tT: SEAT TAKEN\n\n");

printf("Registeration Details:\n");

switch(ch)

{

case 1: printf("Mumbai to Delhi \n");

break;

case 2: printf("Mumbai to Hyderabad \n");

break;

case 3: printf("Mumbai to Kerala \n");

break;

case 4: printf("Mumbai to Ahmedabad \n");

break;

case 5: printf("5.Mumbai to Shrinagar \n");

break;

}

for(i=0;i<n;i++)

{

printf("Name: %s\nPhone No.: %s\nSeat Booked: %s\nDeparture Date: %s\nArrival Date: %s\n",c[i].name,c[i].phonenumber,c[i].seats,dep,arr);

printf("\n");

}

}

void add(int ch)

{

int i,n,j,d,k=0;

int a[10]={0};

printf("Enter the no. of tickets to be booked:\n");

scanf("%d",&n);

printf("Please enter your departure date: (dd/mm/yyyy)\n ");

scanf("%s",&dep);

printf("Please enter your arrival date: (dd/mm/yyyy)\n");

scanf("%s",&arr);

for(i=0;i<n;i++){

printf("Enter your name: \n");

scanf("%s",&c[i].name);

printf("Enter your phone number: \n");

scanf("%s",&c[i].phonenumber);

printf("Pick the seat number you want:\n ");

seat\_arrangement();

printf("\n\t\tE: SEAT EMPTY\t\t\tT: SEAT TAKEN\n\n");

printf("Enter seat number: \n");

scanf("%s",&c[i].seats);

d=atoi(c[i].seats);

strcpy(seats[d],"T");

for(j=0;j<k;j++)

{

if(a[j]==d){

printf("\nSeat Already Taken. Please Select Another One.\n");

scanf("%s",&c[i].seats);

d=atoi(c[i].seats);

strcpy(seats[d],"T");

}

}

a[k++]=d;

}

seat\_arrangement();

display(n,ch);

}

void booking\_dom()

{

int ch,n,seats;

char flight[10];

printf("1.Mumbai to Delhi \n");

printf("2.Mumbai to Hyderabad \n");

printf("3.Mumbai to Kerala \n");

printf("4.Mumbai to Ahmedabad \n");

printf("5.Mumbai to Shrinagar \n");

printf("Enter your choice : \n");

scanf("%d",&ch);

switch(ch)

{

case 1: add(ch);

break;

case 2: add(ch);

break;

case 3: add(ch);

break;

case 4: add(ch);

break;

case 5: add(ch);

break;

}

}

void Domestic()

{

int choice,n;

do{

printf("1. View Flights \n2. Book Tickets \n3. Exit \n");

printf("Enter Your Choice \n");

scanf("%d",&choice);

switch(choice)

{

case 1: printf("1.Mumbai to Delhi \n");

printf("2.Mumbai to Hyderabad \n");

printf("3.Mumbai to Kerala \n");

printf("4.Mumbai to Ahmedabad \n");

printf("5.Mumbai to Shrinagar \n");

printf("Press -1 to exit.");

scanf("%d",&n);

break;

case 2: booking\_dom();

printf("Press -1 to exit.");

scanf("%d",&n);

break;

case 3: exit(0);

}

}while(choice<4);

}

int main()

{

int i,j,index=0;

welcome();

Domestic();

printf("\n");

for(i=0; i<10; i++)

{

printf("\t\t");

for(j=0; j<10; j++)

{

printf("%d.%s\t",index+1,seats[index+1]);

index++;

}

printf("\n");

}

return 0;

}

OUTPUT :

